Amendments

In the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application.

1. (currently amended) A method of synchronizing a plurality of simulation modules, comprising the steps of:

(a) issuing a clock credit to each simulation module;

(b) creating a master clock signal;

(c) dividing the master clock signal to derive an additional clock signal;

(d) applying the additional clock signal to one of the plurality of simulation modules;

[[(b)]] (e) execution, by each simulation module, to an extent corresponding to the clock credit;

[[(c)]] (f) for each simulation module, halting execution when the extent of execution corresponding to the clock credit has been completed; and

[[(d)]] (g) when additional processing by at least one simulation module is necessary, issuing a further clock credit to each simulation module.

2. (original) The method of claim 1, wherein step (a) comprises the step of issuing clock credit to each simulation module on the basis of synchronization points identified in data passing between the simulation modules.

3. (cancelled)

4. (currently amended) The method of claim [[3]] 1, wherein said steps (b), (c), and (d) ~~(e), (f), and (g)~~ are performed by a test bench component of the simulation module.

5. (currently amended) The method of claim 4, further comprising, before said step (b) [[(e)]], the step of:

(h) creating a test bench component for the simulation module.

6. (cancelled)

7. (cancelled)

8. (currently amended) A system for synchronizing a plurality of simulation modules, comprising:

a clock arbitrator that comprises a shared memory interface;

a programming language interface (PLI) for each simulation module, wherein said PLI receives a clock credit from said clock arbitrator and enables and halts simulation module execution on the basis of said clock credit; and

a test bench component for each simulation module, wherein said test bench component manages inputs and outputs to a device under test (DUT) within each simulation module.

9. (cancelled)

10. (cancelled)

11. (amended) A computer program product comprising a computer usable medium having computer readable program code that enables a computer to synchronize a plurality of simulation modules, said computer readable program code comprising:

first computer readable program code logic for causing the computer to issue a clock credit to each simulation module;

second computer readable program code logic for causing the computer to execute each simulation module to an extent corresponding to the clock credit;

third computer readable program code logic for causing the computer to halt execution of a simulation module when the simulation module has completed execution to an extent corresponding to the clock credit; [[and]]

fourth computer readable program code logic for causing the computer to issue a further clock credit to each simulation module to perform additional execution by at least one simulation module;

fifth computer readable program code logic for causing the computer to create a master clock signal;

sixth computer readable program code logic for causing the computer to divide the master clock signal to derive an additional clock signal; and

seventh computer readable program code logic for causing the computer to apply the additional clock signal to one of the plurality of simulation modules.

12. (original) The computer program product of claim 11, wherein said first computer readable program code logic comprises logic for causing the computer to issue clock credit to each simulation module on the basis of synchronization points identified in data passing between the simulation modules.

13. (cancelled)

14. (cancelled)

15. (cancelled)

16. (new) A system for synchronizing a plurality of simulation modules, comprising:

a clock arbitrator;

a programming language interface (PLI) for each simulation module, wherein said PLI receives a clock credit from said clock arbitrator and enables and halts simulation module execution on the basis of said clock credit; and

a test bench component for each simulation module, wherein said test bench component manages inputs and outputs to a device under test (DUT) within each simulation module, and said test bench component and said DUT are compiled together as a single binary executable module.